NAVAL POSTGRADUATE SCHOOL
Monterey, California

MCTSSA Software Reliability Handbook

Volume II

Data Collection Demonstration and
Software Reliability Modeling for a
Multi-Functional Distributed System

by

Norman F. Schneidewind

15 August 1997

DTIC QUALITY INSPECTED 8

19970918 140

NAVAL POSTGRADUATE SCHOOL
Monterey, California
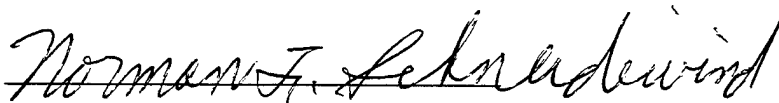
RADM M.J. Evans   Richard Elster
Superintendent   Provost

This report was prepared by:

Norman F. Schneidewind
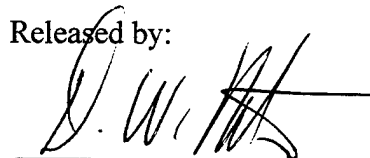
Norman F. Schneidewind
Department of Systems Management

Reviewed by: Released by:

Reuben T. Harris   *D. W. Netzer*

Reuben T. Harris, Chairman   D.W.Netzer, Associate Provost and
Systems Management Department   Dean of Research

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 15 August 1996	3. REPORT TYPE AND DATES COVERED Technical Report

4. TITLE AND SUBTITLE
MCTSSA Software Reliability Handbook,
Volume II
Data Collection Demonstration and Software Reliability Modeling for a Multi-Function Distributed System

5. FUNDING
RLACH

6. AUTHOR(S)
Dr. Norman F. Schneidewind

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Department of Systems Management
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

NPS-SM-97-003

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Marine Corps Tactical Systems Support Activity
Box 555171 Building 31345
Camp Pendleton, CA 92255-5171

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this report are those of the authors and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words.)
The purpose of this handbook is threefold. Specifically, it:
o Serves as a reference guide for implementing standard software reliability practices at Marine Corps Tactical Systems Support Activity and aids in applying the software reliability model
o Serves as a tool for managing the software reliability program
o Serves as a training aid

This handbook consists of four volumes. The content of each of the volumes is as follows:
Volume I: Software Reliability Engineering Process and Modeling for a Single Function System
Volume II: Data Collection Demonstration and Software Reliability Modeling for a Multi-Function Distributed System
Volume III: Integration of Software Metrics with Quality and Reliability
Volume IV: Schneidewind Software Reliability and Metrics Models Tool List

14. SUBJECT TERMS

15. NUMBER OF PAGES
44

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT SAR

NSN 7540-01-280-5800

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std 239-18

MCTSSA SOFTWARE RELIABILITY HANDBOOK

VOLUME II

DATA COLLECTION DEMONSTRATION and SOFTWARE RELIABILITY MODELING FOR a MULTI-FUNCTION DISTRIBUTED SYSTEM

15 August 1996

Revised: 15 July 1997

Dr. Norman F. Schneidewind

Naval Postgraduate School
Code SM/Ss
Monterey, California 93943

Voice: 408-656-2719
Fax: 408-656-3407

Email: schneidewind@nps.navy.mil

TABLE OF CONTENTS

BACKGROUND

During FY95 a Software Reliability Engineering (SRE) handbook and a training plan were developed for the Marine Corps Tactical System Support Activity (MCTSSA) by the Naval Postgraduate School (NPS). This handbook contained a process for the Marine Corps to implement and operate SRE in its software development, test, and operational evaluation activities. The purpose of SRE is to improve the reliability of fielded systems and to treat today's Multi-Function Distributed Systems (MFDS) in a realistic manner for the purpose of software reliability modeling and prediction.

SCOPE AND OBJECTIVES

This handbook applies only to software defects and failures and system failures that are caused by software failures. Hardware failures are excluded. Also the model only includes predictions of software reliability and predictions of system reliability that are based on predictions of software failures [SCH97b, SCH96]. Predictions of hardware reliability are excluded. Interestingly, in the Marine Corps' LOGAIS system (a logistical system to support amphibious operations), which is used as an example application, 4084 (88.3 percent)of the 4584 defects were attributed to software [HEI96]. In this handbook, "user" refers to the user of this handbook.

The objectives of the handbook are to provide a guide for the user to accomplish the following functions:

o   Understand the need for a MFDS model.

o   Understand the terminology of SRE as applied to a MFDS.

o   Understand the structure of the MFDS model.

o   Collect, analyze, and classify defect and failure data.

o   Specify software reliability requirements for a MFDS.

o   Make software reliability predictions for a MFDS.

o   Interpret the results of reliability predictions.

o   Make decisions about software reliability and system reliability (e.g., the system is ready to deploy or, conversely, it requires more testing).

The following major computations are made by the user:

o   Node Failure Probabilities: Boolean search and count operations performed on the defect database.

o   Time to Failure Prediction: Automated in the *SMERFS* tool [MHB96].

o   System Failure Probability: Automated in the *Statgraphics* tool [MHB96].

NEED FOR A MFDS MODEL

Popular software reliability models treat software as a single entity and model the failure process in accordance with this perspective. However in a MFDS, with multiple clients and servers, this approach is not applicable. Consequently a software reliability model was developed that takes into account the fact that not all software defects and failures result in *system failures* in a client-server system. In this model there are critical clients and servers: clients and servers with critical functions (e.g., network communication) that must be kept operational for the system to survive. There are also non-critical clients and servers with non-critical functions (e.g., email). These clients and servers also act as backups for critical clients and servers, respectively. The system does not fail unless all non-critical clients fail and one or more critical clients fail, or all non-critical servers fail and one or more critical servers fail.

MCTSSA required the development of such a model because the MFDS is the type of system that is developed by this agency, where valid predictions of software reliability are important for evaluating the reliability of systems that will be deployed in the field. In addition to the development of a prediction model, it was important to develop an approach to specifying software reliability requirements for client-server systems. These requirements must be stated in terms that recognize the difference between critical and non-critical functions and that a **software defect leading to a software failure does not necessarily result in a system failure**. Furthermore the prediction methodology and the approach for specifying software reliability requirements must be consistent.

The first version of this model, which was developed in FY95, included critical and non-critical clients but did not make this distinction for servers; all servers were treated as critical. The latest version, developed in FY96, eliminates this restriction to allow for non-critical servers. Furthermore, when making reliability predictions, no distinction was made regarding the severity of software defects and failures. The current version categorizes the defects according to severity and its effect on the occurrence of software and system failures and makes predictions for each category. This modification has resulted in a significant increase in prediction accuracy and also provides the software manager with better visibility of software quality as the software is being developed and tested. This was accomplished despite the fact that the *LOGAIS* database -- the source of defect data for this project -- does not contain true software failure data (i.e., failures recorded in CPU time or calendar time). The enhanced model replaces the previous model.

CLIENT-SERVER SOFTWARE RELIABILITY PREDICTION

In order to apply this handbook effectively, it is important that the user first understand the principles of client-server software reliability prediction and have a firm grasp of the terminology that is used in this field. This section provides an introduction to client-server software reliability prediction and provides definitions of several important terms. Too often the assumption is made, when doing software reliability modeling and prediction, that the software involves a *single* node. The reality in today's increasing use of *multi* node client-server systems is that there are multiple entities of software that execute on multiple nodes that must be modeled in a *system* context, if realistic reliability predictions and assessments are to be made. For example if there are N_c clients and N_s servers in a client-server system, it is not necessarily the case that a software failure in any of the N_c clients or N_s servers , which causes the node to fail, will cause the *system* to fail. Thus, if such a system were to be modeled as a single entity, the predicted reliability would be much lower than the true reliability because the prediction would not account for *criticality* and *redundancy*. The first factor accounts for the possibility that the survivability of some clients and servers will be more critical to continued *system* operation than others, while the second factor accounts for the possibility of using redundant nodes to allow for system recovery should a critical node fail. To address this problem, you must identify which nodes -- clients and servers -- are critical and which are not critical. Use the following definitions:

Node: A hardware element on a network, generally a computer, that has a network interface card installed [NOV95].

Client: A node that makes requests of servers in a network or that uses resources available through the servers [NOV95].

Server: A node that provides some type of network service [NOV95]

Client-Server Computing: Intelligence, defined either as processing capability or available information, is distributed across multiple nodes. There can be various degrees of allocation of computing function between the client and server, from one extreme of an application running on the client but with requests for data to the server to the other extreme of a server providing centralized processing (e.g., mail server) and sharing information with the clients [NOV95]. The terms *client-server computing* and *distributed system* are used synonymously.

Critical function: An application function that must operate for the duration of the mission, in accordance with its requirement, in order for the system to achieve its mission goal (e.g., the requirement states that a military field unit must be able to send messages to headquarters and receive messages from headquarters

during the entire time that a military operation is being planned). This type of function operates in the *network mode,* which means that the application requires more than a single client to perform its function; thus client to server or client to client communication is required.

Non-critical function: An application function that does not have to operate for the duration of the mission in order for the system to achieve its mission goal (e.g., it is not necessary to perform word processing during the entire time that a military operation is being planned). Often this type of function operates in the *standalone mode,* which means that a single client performs the application function; thus client to server or client to client communication is not required, except for the possible initial downloading of a program from a file server or the printing of a job at a print server.

Critical clients and servers: Nodes with critical functions, as defined above. These nodes must be kept operational for the system to survive, either by incurring no failures or by reconfiguring non-critical nodes to operate as critical nodes.

Non-critical clients and servers: Nodes with non-critical functions, as defined above. These nodes also act as backups for the critical nodes, should the critical nodes fail.

Software Defect: *Any* undesirable deviation in the operation of the software from its intended operation, as stated in the software requirements.

Software Failure: A defect in the software that *causes* a node (either a client or a server) in a client-server system to be unable to perform its required function within specified performance requirements (i.e., a node failure).

System Failure: The state of a client-server system, which has experienced one or more node failures, wherein there are insufficient numbers and types of nodes available for the system to perform its required functions within specified performance requirements.

CLIENT-SERVER SOFTWARE RELIABILITY SPECIFICATION

In addition to the importance of modeling the correct system configuration when making reliability predictions, it is equally important to state software reliability requirements for a client-server system that are meaningful for the actual operational mode of the system. This section introduces the handbook user to the subject of client-server software reliability specification. Typically, reliability requirements are stated as .999..... What does this mean in operational terms? Technically, according to the IEEE Standard Glossary on Software Engineering Terminology [IEE90], it means there is a .999 "probability that an item will perform a required function under stated conditions for a stated period of time". What is the "item" in the context of a client-server system? The IEEE definition suggests a single entity. In the definition, "function"

is singular, whereas in a client-server system there are multiple functions. How do you operationalize the requirement of .999? Does it mean that a client should be able to execute a given function 99.9 percent of the attempts? How critical is this function relative to other functions? How do you allocate .999 among the various clients and servers?

The specification of software reliability for an MFDS must address the following: 1) definition of critical and non-critical functions; 2) definition of what constitutes "success" and "failure" in executing the functions; 3) consequences of failure to execute these functions correctly; 4) sequence of function execution; and 5) elapsed time in which functions must be completed. With this type of specification in hand, the user can map it to a client-server architecture with a definition of the software and node failure states that would cause a *system* failure.

MODEL FORMULATION

By defining *System Nodes*, *Node Failure Probabilities*, and *Failure States*, the user will be able to compute the probability of system failure *given* that a node failure has occurred. Start by defining the number and type of MFDS nodes as follows:

System Nodes

N_{cc}: Number of Critical Client nodes.

$N_{nc}(t)$: Number of Non-Critical Client nodes.

N_{cs}: Number of Critical Server nodes.

$N_{ns}(t)$: Number of Non-Critical Server nodes.

The sum of these nodes should equal the total number of nodes:

$$N(t)=N_{cc}+N_{nc}(t)+N_{cs}+N_{ns}(t). \tag{1}$$

As long as the system survives, N_{cc} and N_{cs} are constants because a failure of a critical node will result in a non-critical node replacing it, if there is a non-critical node available. A change in software configuration may be necessary on the former non-critical node in order to run the failed critical node's software. If a critical node fails, the system fails, *if there are no non-critical nodes available* on which to run the failed critical node's software.

In contrast, $N_{nc}(t)$ and $N_{ns}(t)$ are decreasing functions of operating time because these nodes replace failed critical nodes, and are not themselves replaced, where $N_{nc}(0)$ is the number of non-critical clients and $N_{ns}(0)$ is the number of non-critical servers at the start of system operation, respectively. In addition, if a non-critical node fails, the function that had been operational on the failed node can be continued on another node of this type and the system can continue to operate in a degraded state. When either a non-critical node

8

replaces a critical node or a non-critical node fails, $N_{nc}(t)$ or $N_{ns}(t)$ is decreased by one, as appropriate.

Node Failure Probabilities

The user must also account for the following node failure probabilities:

p_{cc}: probability of a software defect causing a critical client node to fail.

p_{nc}: probability of a software defect causing a non-critical client node to fail.

p_{cs}: probability of a software defect causing a critical server node to fail.

p_{ns}: probability of a software defect causing a non-critical server node to fail.

These probabilities are important to know individually in the analysis; they are also important in the computation of the probability of *system failure*.

The general function for the probability of system failure by time t, *given a node failure*, is the following:

$$P_{sys}/\text{node fails } (t) = f(N_{cc}, p_{cc}, N_{nc}, p_{nc}, N_{cs}, p_{cs}, N_{ns}, p_{ns}) \tag{2}$$

Equation (2) means that the probability of a system failure, *given a node failure*, is dependent on the four node counts and the corresponding four failure probabilities. The four probabilities are computed from data that is derived from a defect database (defect descriptions, defect classifications, and administrative information) as follows:

$$p_{cc}=\sum_i f_{cc}(i)/D, \text{ where } f_{cc}(i) \text{ is the critical client node failure count in interval } i; \tag{3}$$

$$p_{nc}=\sum_i f_{nc}(i)/D, \text{ where } f_{nc}(i) \text{ is the non-critical client node failure count in interval } i; \tag{4}$$

$$p_{cs}=\sum_i f_{cs}(i)/D, \text{ where } f_{cs}(i) \text{ is the critical server node failure count in interval } i; \tag{5}$$

$$p_{ns}=\sum_i f_{ns}(i)/D, \text{ where } f_{ns}(i) \text{ is the non-critical server node failure count in interval } i; \tag{6}$$

and the total defect count across all intervals is $D=\sum_i d(i)$, $\tag{7}$

where i is the identification of an interval of operating time of the software and d(i) is the total defect count in interval i.

The user makes the computations of equations (3)-(6) by summing the number of failures in a given category (e.g., critical clients) and dividing by equation (7), the total *software defect count* in the database. The *LOGAIS* Chronological Defect Count Database is shown in Appendix A. This table is excerpted from the entire *LOGAIS* database to show the essential data necessary for software reliability analysis. The *LOGAIS* Chronological Node Failure Count Database is shown in Appendix B. This table is derived by querying the database to find those defects in Appendix A that qualify as node failures. Note that there are significantly fewer node failures than there are defect counts, for a given day. Also note that the data in Appendices A and B are not true failure data because the defects and failures are not recorded in CPU execution time or wall clock time. Rather they are recorded in calendar time in batches, based on

9

administrative convenience. Many of these batches are submitted at the end of a workday. This time becomes the "submit date". The details of making these computations will be shown in the *Application of the Model* section.

In a specific application, Boolean expressions (i.e. expressions containing *AND*, *OR*, and *NOT*, logic operations) are used to search the defect database and extract the failure counts (e.g., $_{FCC}(I)$) that are used to compute equations (3)-(6). These expressions specify the conditions that qualify a defect as a node failure (e.g., defect that is a General Protection Fault that affects network operations on a Windows-based system).

Failure States

Next the user needs to know that at a given instant in test or operational time t, a MFDS may be in one of three failure states that pertains to the survivability of the system, as follows, in decreasing order of capability:

Degraded - Type 1: A software defect in a non-critical node causes the node to fail. As a result, the system operates in a degraded state, with one less non-critical node. No reconfiguration is necessary because the failed node is not replaced.

Degraded - Type 2: A software defect in a critical node causes the node to fail. As a result, the system operates in a degraded state, but one that is more severe than *Type 1*, because there would be both a temporary loss of one critical node during reconfiguration and a permanent loss of one non-critical node (i.e., one of the non-critical nodes takes over the function of the failed critical node). Under certain conditions -- see Table 1 -- this type of node failure can cause a system failure.

The current version of the model assumes that node failures are not recoverable on the node where the failure occurred, *during the mission*. The next version of the model will contain a repair function to account for the case where a node failure is repaired and the node is put back into operation during the mission.

System Failure: The system fails under the following conditions: 1) all non-critical clients fail **and** one or more critical clients fail, **or** 2) all non-critical servers fail **and** one or more critical servers fail. The reason for this failure event formulation is that, in the event of a failed critical node, a non-critical node can be substituted, possibly with a different software configuration. However, if all non-critical clients (servers) fail, and one or more critical clients (servers) fail, there would be no non-critical clients (servers) left to take over for the failed critical clients (servers).

10

The failure states are summarized in Table 1.

Table 1
Failure States

	Degraded - Type 1	Degraded - Type 2	System Failure
Non-Critical Client	Node Failure	Does Not Apply	Does Not Apply
Critical Client	Does Not Apply	Node Failure(s) and $N_{nc}(t)>0$	Node Failure(s) and $N_{nc}(t)=0$
Non-Critical Server	Node Failure	Does Not Apply	Does Not Apply
Critical Server	Does Not Apply	Node Failure(s) and $N_{ns}(t)>0$	Node Failure(s) and $N_{ns}(t)=0$

System Failure Probability

Having equations (3)-(6) for the node failure probabilities in hand, the model applies them to computing the probability of system failure -- equation (12). The intermediate equations leading up to equation (12) follow:

The probability that **one or more** critical clients N_{cc} fail, given that the software fails, is:

$$P_{cc}=1-(1-p_{cc})^{Ncc} \qquad (8)$$

The probability that **all** non-critical clients $N_{nc}(t)$ have failed by time t, given that the software fails, is:

$$P_{nc}(t)=(p_{nc})^{Nnc(t)} \qquad (9)$$

The probability that **one or more** critical servers N_{cs} fail, given that the software fails, is:

$$P_{cs}=1-(1-p_{cs})^{Ncs} \qquad (10)$$

The probability that **all** non-critical servers $N_{ns}(t)$ have failed by time t, given that the software fails, is:

$$P_{ns}(t)=(p_{ns})^{Nns(t)} \qquad (11)$$

Equations (8) and (9) assume that client failures are independent (i.e., one type of node failure does not cause another type of node failure). This is the case because a failure in one client's software would not cause a failure in another client's software. However it is possible that a failure in server software could cause a failure in client software, such as a client accessing a server that has corrupted data. Also, equations (10) and (11) assume that server failures are independent. This is the case because a failure in one server's software would not cause a failure in another server's software. However it is possible that a failure in client software could cause a failure in server software, such as a client with corrupted data accessing a server. No case of client failures that were caused by server failures nor of the converse have been found in the *LOGAIS* database. Of course, this does not mean that these events could not happen in general. To account for the

11

possibility of these events, you would need to include the conditional probability of a client failure, given a server failure, and the converse. This model formulation is beyond the scope of this handbook and will be included in the next version of the model.

Combining (8), (9), (10), and (11), the probability of a system failure by time t, given that a node fails, is:

$$P_{sys}/\text{node fails } (t)=[P_{cc}][P_{nc}(t)]+[P_{cs}][P_{ns}(t)]=[1-(1-p_{cc})^{Ncc}][(p_{nc})^{Nnc(t)}]+[1-(1-p_{cs})^{Ncs}][(p_{ns})^{Nns(t)}] \quad (12)$$

and the probability of a node failure due to software is:

$$p_{sw}=p_{cc}+p_{nc}+p_{cs}+p_{ns} \quad (13)$$

Note that the user is not required to make the computations for equations (8)-(11), which are shown for informational and terminology purposes. The user only computes equation (12), which itself is automated as will be shown in the *Application of the Model* section.

Model Concepts

The model concepts are illustrated in Figures 1 and 2, where there are five critical clients, five non-critical clients, one critical server, and one non-critical server. Figure 1 shows a surviving configuration, where a critical client fails and a critical server fails but there are non-critical clients and a non-critical server to take over the functions of the failed nodes. The consequence of this configuration is a *Degraded - Type 2* failure state. Figure 2 shows a failing configuration where there are no non-critical clients and server to take over for the critical failing nodes. The consequence of this configuration is a *system failure*. In both figures, for illustrative purposes, both a failed critical client and a failed critical server are shown. A more typical case is when only one of the critical nodes fails at a time.
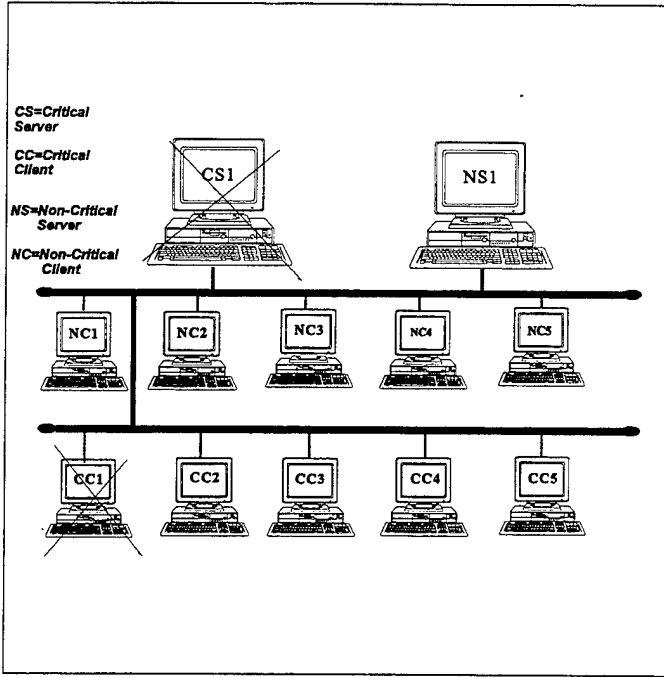
Figure 1. Surviving Configuration
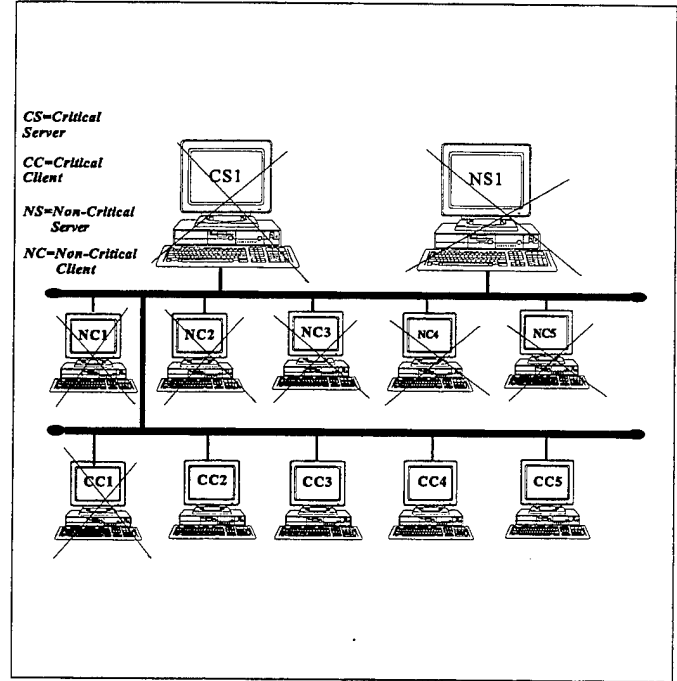
Figure 2. Failing Configuration

Time to Failure Prediction

In order to make *Time to Failure* predictions for each of the four types of node failures, the user first analyzes the defect data to determine what type of software defects could cause each of the four types of node failures; then the user partitions the defect data accordingly. More will be said about this process in the *Application of Model* section. Next the user applies equation (14) of the *Schneidewind Software Reliability Model* [AIA93, KEL95, LYU 96, SCH97a, SCH93, SCH92] to make each of the four predictions, using the *SMERFS* software reliability tool [FAR93]. In equation (14), $T_f(t)$ is the predicted time (intervals) until the next F_t failures (one or more) occur, α and β are failure rate parameters, s is the first interval where the observed failure data is used, t is the current interval, and $X_{s,t}$ is the cumulative number of failures observed in the range s,t.

$$T_F(t)=[(\log[\alpha/(\alpha-\beta(X_{s,t}+F_t)])/\beta]-(t-s+1)$$

$$\text{for } (\alpha/\beta)>(X_{s,t}+F_t)$$

(14)

Time to Failure predictions are made for critical clients, non-critical clients, critical servers, and non-critical servers. As the predicted failure times are recorded, the user observes whether the condition for system failure, as defined previously, has been met. If this is the case, a predicted system failure is recorded.

13

Thus, in addition to monitoring the types of predicted failures (e.g., critical client), the process also involves monitoring $N_{nc}(t)$ and $N_{ns}(t)$ to identify the time t when either is reduced to zero, signifying that the supply of non-critical clients or non-critical servers has been exhausted. In this situation, a failure of a critical client or critical server, respectively, will result in a system failure. Thus the user predicts a system failure when the following expression is true (where "$\wedge$" means "AND" and "$\vee$" means "OR"):

((Predict critical client failure)$\wedge$($N_{nc}(t)$=0))$\vee$((Predict critical server failure)$\wedge$($N_{ns}(t)$=0))   (15).

If the predictions produce multiple node failures in the same interval (e.g., critical client and critical server), the user records multiple failures for that interval.

APPLICATION OF THE MODEL

Analysis of the Defect and Failure Data

In this example the user applies the software reliability model to the Marine Corps LOGAIS system -- a client-server logistical support system. In this system it is important that the reliability specification distinguish between failure states *Degraded-Type 1*, *Degraded-Type 2*, and *System Failure*, as previously defined (i.e., distinguish between node failures that cause performance degradation but allow the system to survive, and node failures that cause a system failure). This distinction is made when analyzing the system's defect data. The defect data used in the example are from the LOGAIS defect database, using the *Defect Control System* (DCS), a defect database management system which was used on the *LOGAIS* project [MHB96, MTP96]. The network configurations in Figures 1 and 2 are used in this example.

In this Windows-based client-server system, the types of clients and servers that were previously defined are used, with corresponding types of defects and failures, as identified in the defect database [MHB96, MTP96]. The following short-hand notation for identifying the attributes of the defect database is used:

o S: Software Defect

o G: General Protection Fault (GPF)

o N: Network Related Failure

o C: System Crash

The LOGAIS defect database is queried in order to identify the software defects that qualify as node failures. The following Boolean expressions, corresponding to the four types of node failures, are used:

1. Critical Client Failure: COUNT as failures WHERE (S$\wedge$G$\wedge$N$\wedge$*notC*). A *GPF* causes a node failure (*Degraded-Type 2*) on a critical client, a client which must maintain communication with other nodes on the network (*Network Mode*), and the failure does not cause a *System Crash* (loss of server).

14

2. Non-Critical Client Failure: COUNT as failures WHERE ($S \wedge G \wedge notN \wedge notC$). A *GPF* causes a node failure (*Degraded-Type 1*) on a non-critical client, a client which does not have to maintain communication with other nodes on the network (*Standalone Mode*), and the failure does not cause a *System Crash* (loss of server).

3. Critical Server Failure: COUNT as failures WHERE ($S \wedge notG \wedge N \wedge C$). A *System Crash* causes a node failure (*Degraded-Type 2*) on a critical server, a server which must maintain communication with other nodes on the network (*Network Mode*), and the failure is not a *GPF;* it is more serious, resulting in the loss of a server.

4. Non-Critical Server Failure: COUNT as failures WHERE ($S \wedge notG \wedge notN \wedge C$). A *System Crash* causes a node failure (*Degraded-Type 1*) on a non-critical server, a server which does not have to maintain communication with other nodes on the network, and the failure is not a *GPF;* it is more serious, resulting in the loss of a server.

The above classification associates *GPF* with clients and *System Crash* with servers; it also associates *Network Related Failures* with critical node failures. Note that this is only an example. For other systems, different defect and failure classifications may be appropriate.

The total failure count is obtained by taking the union of expressions 1--4 as follows:

5. Total Failure Count: COUNT as failures WHERE ($S \wedge ((G \wedge notC) \vee (notG \wedge C))$). This expression is used to verify the correctness of 1--4 because it should equal their sum.

An example of querying the *LOGAIS* database for critical client defects that constitute node failures, using DCS, is shown in Figures 3 and 4. The first figure shows the selection criteria; the second figure shows the defect record of the first defect that satisfies the criteria: Defect # 1401, which occurred in Interval 16 on 11/29/94 (see Appendix B). Appendix B is created by querying the defect database, using expressions 1--4, and counting the occurrences that satisfy the criteria. Note that Appendix B (and Appendix A) are created only once; they are updated as new defects and failures occur.

Observed Range and Prediction Range

The major objective of reliability modeling is to predict future reliability over the prediction range of test or operational time of a system. However to do so, there must be a historical record of defects and failures for computing the model parameters and for making the best fit with the historical data; the data is collected during the observed range of test or operational time of a system. The length of the observed range is determined by the amount of data that has been collected prior to making a prediction, while the length of the prediction range is determined by duration of the system's mission. The observed range in this

15

example is 1,50 intervals and the prediction range is 51,61 intervals for the data in Appendix B. These ranges are arbitrary and selected only to illustrate the process. The user should note that once a system has been tested or operated over the prediction range, there will be observed defects and failures in this range. The observed defects and failures in the prediction range are listed in Table 2. The failure counts corresponding to types 1--5, above, are summarized in Table 3, which shows the empirical probabilities of node failure that are computed using equations (3)--(7) and (13). For example, for critical clients, the computation is 24/4048=.005929. The user should verify the computations for the remaining types of nodes.

Figure 3. Defect Control System Query Menu for Critical Clients

File   Edit   Submit   Notify   Update   Query   Report   Admin   Tools   Help

Standard   Critical

Id	Submitter	Owner Id	Title
d-1401	3735	una	9.67 CAEMS Table and UDL actions

Status:	Closed	**More Symptoms:**	<<None>>
Symptoms:	GPF	**System:**	COMMON
Net/Stand Alone:	On Network	**How submitted:**	DCS
How Found:	Testing	**Main Module:**	<<None>>
Hardware:	486 Desktop	**SubModule:**	<<None>>
Company:	<<None>>	**Operating System:**	MS DOS 6.2/Windows 3.11
Priority:	HIGH	**Category:**	Software
Resolution:	<<None>>		
Related Reports:			
Build Version:			

Description: had created a new record in UDL, was attempting to copy (had entire record highlighted once then redid with only part of record highlighted after rebooted), selected "Cut Records" by accident, got GPF caused by PBSTUB in Module PBRTE030.DLL at 4C:048F

Ready   8/13/96   10:10:26 AM   Query: 1 of 1   Total Defects: 4584

Figure 4. Defect Control System Query Result for Critical Clients

Table 2

LOGAIS Chronological Node Failure Count Database (Sample)

CC: Critical Client Node Failure

NC: Non-Critical Client Node Failure

CS: Critical Server Node Failure

NS: Non-Critical Server Node Failure

Interval	Defect ID	Number	Submit Date	CC	NC	CS	NS
51	2633,2634	2	1/24/95		X		
51	2635,2636,2637,2638	4	1/24/95				X
52							
53	2661,2662,2663,2664	4	1/26/95		X		
54	2641,2644,2645,2669, 2671,2672,2673,3003	8	1/27/95		X		
54	2640,2643,2670,2674, 2675,2676,2783	7	1/27/95				X
55	2450	1	1/30/95		X		
56							
57							
58	2487	1	2/2/95				X
59	2511,2512,2513	3	2/3/95		X		
60							
61	3025,3026,3027,3029	4	2/7/95	X			

Table 3

Summary of Node Failures (4048 Software Defects)

	Number of Failures	Probability
1. Critical Client	24	p_{cc}=.005929
2. Non-Critical Client	83	p_{nc}=.020250
3. Critical Server	2	p_{Cs}=.000494
4. Non-Critical Server	158	p_{ns}=.039032
5. Total	267	p_{sw}=.065705

Application Predictions

Time to Failure

Using equation (14) and failure data in the observed range 1,50 (see Appendix B), the user makes predictions for *Time to Failure*, for $t>50$ days, for critical clients. The predictions are made for a given numbers of failures (time to one failure for $t>50$ days, time to two failures for $t>50$ days, etc.). The excerpted *SMERFS* session for these predictions is shown in Appendix C. The user should carefully study this example. In the case of critical clients, the failure data was sparse; thus a five day interval was used for prediction, with these predictions converted to one day intervals as shown in Appendix C. Table 4 shows predictions compared with the actual failure data (obtained from Table 2), with the relative error and average relative error for cumulative values shown. Both *Time to Failure* and *Cumulative Time to Failure* (i.e., cumulative total starting with *50 days*). It is not necessary for the user to make these comparisons; they are shown only to give the user a feel for the accuracy of prediction.

Table 4
Critical Client Predictions Made at Time=50 Days
Observed Range=1,50 Days; Failure Count=11; Prediction Range>50 Days

Predicted			Actual		
Given Number of Failures	Time to Failure (Days)	Cumulative Time to Failure (Days)	Time to Failure (Days)	Cumulative Time to Failure (Days)	Relative Error (Percent)
1	5.19	55.19	11	61	-9.52
2	11.07	61.07	11	61	+.11
3	17.88	67.88	11	61	+11.28
4	25.95	75.95	11	61	+24.51
5	35.86	85.86	36	86	-.16
				Average	9.12%

20

Similarly, predictions are made for non-critical clients and non-critical servers in Tables 5 and 6, respectively. In the case of these non-critical nodes, the failure data is sufficiently dense to allow a failure count interval of one day for predictions. In the case of critical servers, there are only two actual failures, both of which occur in the observed range (see Appendix B). Only one prediction of *Time to Failure* for **one more failure** could be made at t=50 for critical servers because the predicted remaining failures at t=50 is 1.40; therefore, critical server failures are not tabulated. Using *SMERFS*, the user should verify the results in Tables 5 and 6.

Table 5
Non-Critical Client Predictions Made at Time=50 Days
Observed Range=1,50 Days; Failure Count=36; Prediction Range>50 Days

Predicted			Actual		
Given Number of Failures	Time to Failure (Days)	Cumulative Time to Failure (Days)	Time to Failure (Days)	Cumulative Time to Failure (Days)	Relative Error (Percent)
1	2.41	52.41	1	51	+2.76
2	4.87	54.87	1	51	+7.59
3	7.37	57.37	3	53	+8.25
4	9.92	59.92	3	53	+13.06
5	12.52	62.52	3	53	+17.96
				Average	+ 9.92%

Table 6
Non-Critical Server Predictions Made at Time=50 Days
Observed Range=1,50 Days; Failure Count=108; Prediction Range>50 Days

Predicted			Actual		
Given Number of Failures	Time to Failure (Days)	Cumulative Time to Failure (Days)	Time to Failure (Days)	Cumulative Time to Failure (Days)	Relative Error (Percent)
1	1.96	51.96	1	51	+1.88
2	3.93	53.93	1	51	+5.75
3	5.90	55.90	1	51	+9.61
4	7.87	57.87	1	51	+13.47
5	9.84	59.84	4	54	+10.81
				Average	+ 8.30%

Using the data in Tables 4-6, the user merges and sequences the various types of failure **predictions** in Table 7. The purpose of this table is to construct the scenario of failures and surviving non-critical nodes so that the time of *System Failure* can be predicted. The table shows that seven node failures (i.e., the sequence NS, NC, NC, CC, NC, NC, CC) are predicted to occur before the system is predicted to fail. This occurs at t=61.07 days when there are no non-critical clients available and a critical client fails. No critical server failures are shown in this table because the prediction of *Time to Failure* of 99.35 days cumulative is beyond the prediction range of interest in this example. The user should verify, by examining Tables 4-6, that the **predicted** sequence of *Cumulative Time to Failure* and *Type of Failure* occur as shown in Table 7.

Table 7
Predicted Time to Failure When Various Types of Failures are Merged and Sequenced
Observed Range=1,50 Days; Prediction Range=51,61 Days

CC: Critical Client
NC: Non-Critical Client
NS: Non-Critical Server

Cumulative Time to Failure (Days)	Time to Failure (Days)	Type of Failure	Number of Non-Critical Clients Available	Number of Non-Critical Servers Available
50	1.96		5	1
51.96	.45	NS	5	0
52.41	2.46	NC	4	0
54.87	.32	NC	3	0
55.19	2.18	. CC	2	0
57.37	2.55	NC	1	0
59.92	1.15	NC	0	0
61.07		CC	**System Failure**	

Using the data in Table 7, the user plots **predicted** cumulative failures and number of available non-critical clients versus cumulative time to failure in Figure 5. This graph shows the accumulation of node failures, with the corresponding reduction in the available non-critical clients, until the maximum allowable failures occurs, and the system fails.

Using the data in Tables 4-6, the user merges and sequences the various types of **actual** failures in Table

8. Similar to Table 7, the purpose of this table is to construct the scenario of actual failures and surviving non-critical nodes so that the actual time of *System Failure* can be determined and compared with the predicted values. As in the case of the predictions, this table shows that seven node failures (i.e., the sequence NC, NS, NC, NC, NC, NC, CC) occur before the system fails. This occurs at t=61days when there are no non-critical clients available and a critical client fails. No critical server failures are shown in this table because they occurred prior to the range of this example. The user should verify, by examining Tables 4-6, that the **actual** sequence of *Cumulative Time to Failure* and *Type of Failure* occur as shown in Table 8.

Table 8
Actual Time to Failure When Various Types of Failures are Merged and Sequenced
Range=51,61 Days

CC: Critical Client
NC: Non-Critical Client
NS: Non-Critical Server

Cumulative Time to Failure (Days)	Time to Failure (Days)	Type of Failure	Number of Non-Critical Clients Available	Number of Non-Critical Servers Available
50	1		5	1
51	0	NC,NS	4	0
51	2	NC	3	0
53	0	NC	2	0
53	0	NC	1	0
53	8	NC	0	0
61		CC	**System Failure**	

Using the data in Table 8, the user plots **actual** cumulative failures and number of available non-critical clients versus cumulative time to failure in Figure 6. The shape of Figure 6 is caused by multiple failures occurring on the same day in some cases. In comparing Figures 3 and 4, the user will see that in each case seven node failures are required to cause a system failure and that the system fails on Day 61; however, the supply of non-critical clients becomes exhausted earlier in the actual case.

23

Predicted Cumulative Failures and Available Non-Critical Clients

CC: Critical Client, NC: Non-critical Client, NS: Non-critical Server
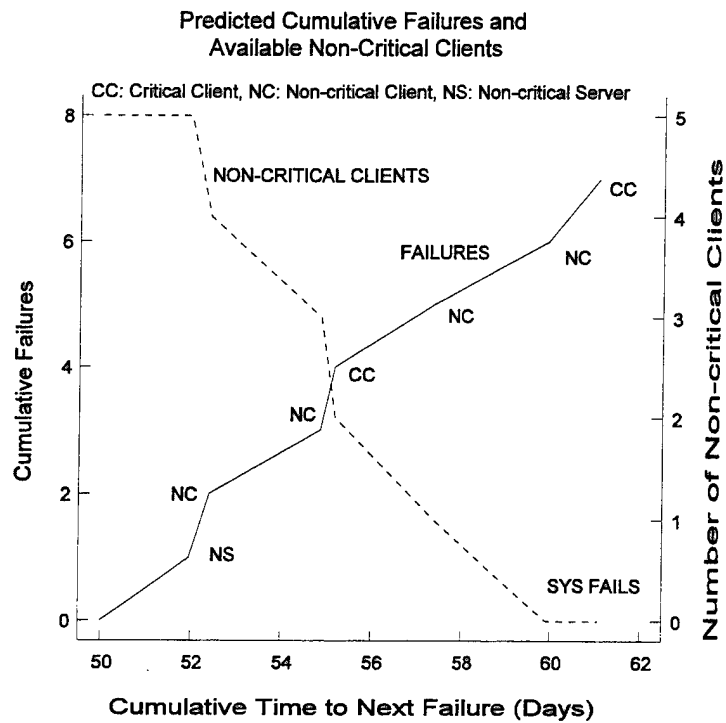
Figure 5. Predicted Node and System Failures

Actual Cumulative Failures and Available Non-Critical Clients

CC: Critical Client, NC: Non-critical Client, Ns: Non-critical Server
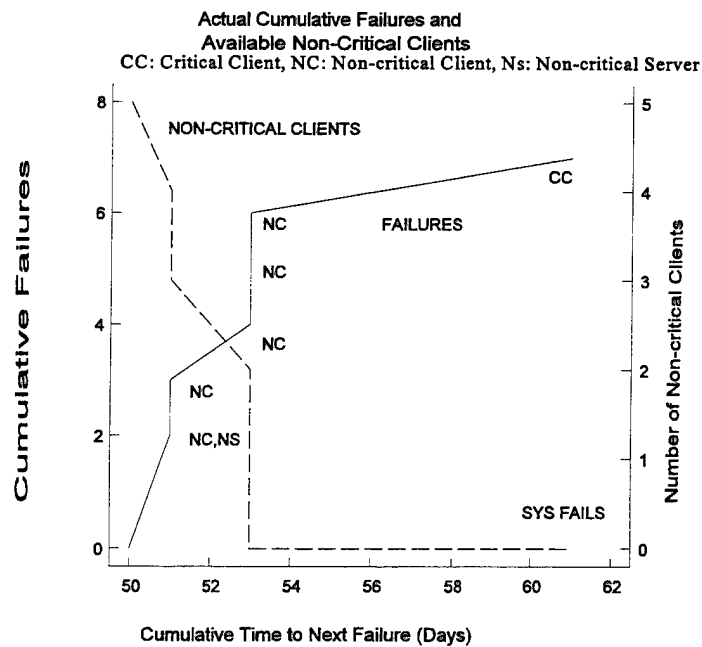
Figure 6. Actual Node and System Failures

24

Probability of System Failure

Lastly, using equation (12), the user predicts the probability of system failure, given a node failure, in column 5 of Table 9, as the system progresses through the predicted failure scenario that was shown in Table 7 and Figure 5. Except for row 2 in Table 9, the actual probability is the same as the predicted probability because the actual failure scenario that was shown in Table 8 and Figure 6 produces the same numbers of non-critical clients and servers that are shown in columns 6 and 7, respectively. Because the predicted and actual failure scenarios are identical, except for row 2, the predicted time to failure and type of node failure, columns 1 and 2, respectively, can be compared in with the corresponding actual values in columns 3 and 4, for given probabilities of system failure. These values were reproduced from Tables 7 and 8, respectively. Because for a given $P_{sys}/node\ fails$ (t), the cumulative time to failure occurs later for the predicted values, the model is a bit optimistic with respect to reality for this example. Note that the in the last row of Table 9 the system has not yet failed. This occurs when a critical client fails at Day 61.07 *predicted* (see Table 7 and Figure 5) and at Day 61 *actual* (see Table 8 and Figure 6). At this time there are no non-critical clients left to replace the failed critical client.

The computation of equation (12) is shown in Appendix D, using Statgraphics. This equation is created and saved by using the equation editor of Statgraphics. Then the various input values are loaded and the equation is evaluated to produce the probabilities in column 5 of Table 9.The user should verify the computations in Appendix D.

The user should note the significant results that emerge from this analysis. They are: 1) The $P_{sys}/node$ *fails* (t) is only significant (.029790) when the supply of both non-critical clients and non-critical servers has been exhausted and 2) $P_{sys}/node\ fails$ (t) is significantly lower than the probability of *any* type of node failure caused by a software defect: p_{sw}=.065705, obtained from equation (13) and computed in Table 3. Thus evaluations of system reliability should recognize that *software failures are not necessarily equivalent to system failures* and that assessments of software reliability that treat every failure as equivalent to a system failure will grossly understate system reliability.

Table 9
Probability of System Failure

Predicted Cumulative Time to Failure (Days)	Predicted Type of Node Failure	Actual Cumulative Time to Failure (Days)	Actual Type of Node Failure	Probability of System Failure Given a Node Failure	Number of Non-Critical Clients Available	Number of Non-Critical Servers Available
50		50		0.000019	5	1
51.96	NS			0.000494*	5*	0*
52.41	NC	51	NC,NS	0.000494	4	0
54.87	NC	51	NC	0.000494	3	0
55.19	CC	53	NC	0.000506	2	0
57.37	NC	53	NC	0.001087	1	0
59.92	NC	53	NC	0.029790	0	0

* Applies only to predicted values.

SUMMARY

This handbook provides a software reliability model for a MFDS that includes both critical clients and servers and non-critical clients and servers. In order to use this model, the user must partition the defects and failures into classes that are then associated with critical and non-critical clients and servers. Once this is done, predictions can be made of *Time to Failure* for each type of node. The predictions are classified according to those that would result in a node failure caused by a software defect and those that would result in a system failure caused by a series of software defects. Then the probability of system failure is computed. The computations for the *LOGAIS* example illustrated a significant principle of MFDS reliability assessment: *software failures should not be treated as the equivalent of system failures because to do so would grossly understate system reliability.*

Possible enhancements of this model include the following: extend the model to include hardware failures; develop measures of performance degradation, as nodes fail; include a node repair rate to reflect the possibility of recovering failed nodes during the operation of the system; and apply smoothing techniques, such as the moving average, to mitigate anomalies in calendar time defect data.

REFERENCES

[AIA93]   Recommended Practice for Software Reliability, R-013-1992, American National Standards Institute/American Institute of Aeronautics and Astronautics, 370 L'Enfant Promenade, SW, Washington, DC 20024, 1993.

[FAR93]   William H. Farr and Oliver D. Smith, Statistical Modeling and Estimation of Reliability Functions for Software (SMERFS) Users Guide, NAVSWC TR-84-373, Revision 3, Naval Surface Weapons Center, Revised September 1993.

[HEI96]   Judie Heineman, Norman Schneidewind, and Kenneth Warburton, "Software Reliability Engineering Process Experience Report", Proceedings of The Eighth Annual Software Technology Conference, Salt Lake City, Utah, 21-26 April 1996, 17 pages.

[IEE90]   IEEE Standard Glossary of Software Engineering Terminology, IEEE Std 610.12.1990.

[KEL95]   Ted Keller, Norman F. Schneidewind, and Patti A. Thornton "Predictions for Increasing Confidence in the Reliability of the Space Shuttle Flight Software", Proceedings of the AIAA Computing in Aerospace 10, San Antonio, TX, March 28, 1995, pp. 1-8.

[LYU96]   Michael R. Lyu (Editor-in-Chief), Handbook of Software Reliability Engineering, Computer Society Press, Los Alamitos, CA and McGraw-Hill, New York, NY, 1995.

[MHB96]   MCTSSA Software Reliability Handbook, Norman F. Schneidewind and Judie A. Heineman, Naval Postgraduate School, January 10, 1996.

[MTP96]   MCTSSA Software Reliability Engineering Training Plan, Norman F. Schneidewind and Judie A. Heineman, Naval Postgraduate School, January 10, 1996.

[NOV95]   Werner Feibel, Novell's Complete Encyclopedia of Networking, Novell Press, San Jose, CA, 1995.

[SCH97a]   Norman F. Schneidewind, "Reliability Modeling for Safety Critical Software", IEEE Transactions on Reliability, Vol. 46, No.1, March 1997, pp.88-98.

[SCH97b]   Norman F. Schneidewind, "Data Collection Demonstration and Software Reliability Modeling for a Multi-Function Distributed System", Proceedings of the Twentieth Annual Software Engineering Workshop, NASA Goddard, Greenbelt, Maryland, March 1997, pp.231-240.

[SCH96]   Norman F. Schneidewind, "Software Reliability Engineering for Client-Server Systems", Proceedings of The Seventh International Symposium on Software Reliability Engineering, White Plains, New York October 30, 1996 - November 2, 1996, pp.226-235.

[SCH93]   Norman F. Schneidewind, "Software Reliability Model with Optimal Selection of Failure Data", IEEE Transactions on Software Engineering, Vol. 19, No. 11, November 1993, pp. 1095-1104.

[SCH92]   Norman F. Schneidewind and T. W. Keller, "Application of Reliability Models to the Space Shuttle", IEEE Software, Vol. 9, No. 4, July 1992 pp. 28-33.

APPENDIX A
LOGAIS Chronological Defect Count Database

Count Interval	Defect ID Range	Number of Defects	Submit Date	Day
1	1-120	120	11/11/94	Fri
2	121-305	185	11/12/94	Sat
3	306	1	11/13/94	Sun
4	307-497	191	11/14/94	Mon
5	498-710	213	11/15/94	Tue
	5 Day Total	**710**		
	Cumulative	**710**		
6	711-888	178	11/16/94	Wed
7	889-942	54	11/17/94	Thu
8	943-981	39	11/18/94	Fri
9	982-996	15	11/19/94	Sat
10	997-1024	28	11/20/94	Sun
	5 Day Total	**314**		
	Cumulative	**1024**		
11	1025-1123	99	11/21/94	Mon
12	1124-1193	70	11/22/94	Tue
13	1194-1253	60	11/23/94	**Wed**
14	1254-1263	10	11/25/94	**Fri**
15	1264-1368	105	11/28/94	**Mon**
	5 Day Total	**344**		
	Cumulative	**1368**		
16	1369-1483	115	11/29/94	Tue
17	1484-1565	82	11/30/94	Wed
18	1566-1624	59	12/1/94	Thu
19	1625-1697	73	12/2/94	Fri
20	1698-1703	6	12/3/94	Sat
	5 Day Total	**335**		
	Cumulative	**1703**		

28

21	1704-1721	18	12/4/94	Sun
22	1722-1740	19	12/5/94	Mon
23	1741-1772	32	12/6/94	Tue
24	1773-1803	31	12/7/94	Wed
25	1804-1823	20	12/8/94	Thu
	5 Day Total	**120**		
	Cumulative	**1823**		
26	1824-1830	7	**12/9/94**	**Fri**
27	1831-1840	10	**12/15/94**	**Thu**
28	1841-1861	21	**12/19/94**	**Mon**
29	1862-1915	54	12/20/94	Tue
30	1916-1929	14	12/21/94	Wed
	5 Day Total	**106**		
	Cumulative	**1929**		
31	1930-1935	6	12/22/94	Thu
32	1936-1960	25	**12/23/94**	**Fri**
33	1961-1964	4	**12/28/94**	**Wed**
34	1965-1982	18	12/29/94	Thu
35	1983-1985	3	**12/30/94**	**Fri**
	5 Day Total	**56**		
	Cumulative	**1985**		
36	1986	1	**1/3/95**	**Tue**
37	1987-2000	14	1/4/95	Wed
38	2001-2003	3	1/5/95	Thu
39	2004-2027	24	**1/6/95**	**Fri**
40	2028-2093	66	1/9/95	Mon
	5 Day Total	**108**		
	Cumulative	**2093**		
41	2094-2157	64	1/10/95	Tue
42	2158-2231	74	1/11/95	Wed
43	2232-2292	61	1/12/95	Thu

44	2293-2358	66	1/13/95	Fri
45	2359-2362	4	**1/14/95**	**Sat**
	5 Day Total	**269**		
	Cumulative	**2362**		
46	2363-2372	10	**1/16/95**	**Mon**
47	2373-2390	18	1/17/95	Tue
48	2391-2399	9	1/18/95	Wed
49	2400-2405	6	1/19/95	Thu
50	2406-2424	19	**1/20/95**	Fri
	5 Day Total	**62**		
	Cumulative	**2424**		
51	2425-***	48	**1/24/95**	**Tue**
52	2426-***	44	1/25/95	Wed
53	2430-***	145	1/26/95	Thu
54	2433-***	227	**1/27/95**	**Fri**
55	2446-2473	28	**1/30/95**	**Mon**
	5 Day Total	**492**		
	Cumulative	**2916**		
56	2474-2480	7	1/31/95	Tue
57	2481-2486	6	2/1/95	Wed
58	2487-2510	24	2/2/95	Thu
59	2511-2529	19	**2/3/95**	Fri
60	2530-2543	14	2/6/95	Mon
	5 Day Total	**70**		
	Cumulative	**2986**		
61	2544***	53	2/7/95	Tue
62	3040-3067	28	2/8/95	Wed
63	3068-3099	32	2/9/95	Thu
64	3100-3110	11	**2/10/95**	**Fri**
65	3111-3137	27	2/13/95	Mon
	5 Day Total	**151**		

	Cumulative	3137		
66	3138-3146	9	2/14/95	Tue
67	3147-3167	21	2/15/95	Wed
68	3168-3213	46	2/16/95	Thu
69	3214-3233	20	2/17/95	Fri
70	3234-3242	9	2/21/95	Tue
	5 Day Total	105		
	Cumulative	3242		
71	3243-3260	18	2/22/95	Wed
72	3261-3314	54	2/23/95	Thu
73	3315-3320	6	2/24/95	Fri
74	3321-3324	4	2/27/95	Mon
75	3325-3334	10	2/28/95	Tue
	5 Day Total	92		
	Cumulative	3334		
76	3335-3340	6	3/1/95	Wed
77	3341	1	3/2/95	Thu
78	3342-3343	2	3/3/95	Fri
79	3344-3347	4	3/6/95	Mon
80	3348-3349	2	3/7/95	Tue
	5 Day Total	15		
	Cumulative	3349		
81	3350-3362	13	3/8/95	Wed
82	3363-3368	6	3/10/95	Fri
83	3369-3379	11	3/13/95	Mon
84	3380-3383	4	3/14/95	Tue
85	3384-3419	36	3/15/95	Wed
	5 Day Total	70		
	Cumulative	3419		
86	3420-3431	12	3/16/95	Thu
87	3432-3447	16	3/17/95	Fri

88	3448-3492	45	3/20/95	**Mon**
89	3493-3530	38	3/21/95	Tue
90	3531-3566	36	3/22/95	Wed
	5 Day Total	**147**		
	Cumulative	**3566**		
91	3567-3601	35	3/23/95	Thu
92	3602-3616	15	3/24/95	**Fri**
93	3617-3635	19	3/27/95	**Mon**
94	3636-3652	17	3/28/95	Tue
95	3653-3658	6	3/29/95	Wed
	5 Day Total	**92**		
	Cumulative	**3658**		
96	3659-3681	23	3/30/95	Thu
97	3682-3693	12	3/31/95	Fri
98	3694-3710	17	4/3/95	**Mon**
99	3711-3726	16	4/4/95	Tue
100	3727-3731	5	4/5/95	Wed
	5 Day Total	**73**		
	Cumulative	**3731**		
101	3732-3769	38	4/6/95	Thu
102	3770-3840	71	4/7/95	**Fri**
103	3841	1	4/10/95	**Mon**
104	3842-3856	15	4/11/95	Tue
105	3857-3885	29	4/12/95	Wed
	5 Day Total	**154**		
	Cumulative	**3885**		
106	3906-***	23	4/13/95	Thu
107	3909-3923	15	4/14/95	**Fri**
108	3924-3932	9	4/16/95	**Sun**
109	3933-3949	17	4/17/95	Mon
110	3950-3963	14	4/18/95	Tue

	5 Day Total	**78**		
	Cumulative	**3963**		
111	3964-4033	70	4/19/95	Wed
112	4034-4079	46	**4/20/95**	**Thu**
113	4080-4107	28	**4/25/95**	**Tue**
114	4108-4132	25	4/26/95	Wed
115	4133-4167	35	4/27/95	Thu
	5 Day Total	**204**		
	Cumulative	**4167**		
116	4168-4176	9	**4/28/95**	**Fri**
117	4177-4185	9	**5/1/95**	**Mon**
118	4186-4207	22	5/2/95	Tue
119	4208-4287	80	5/3/95	Wed
120	4288-4320	33	5/4/95	Thu
	5 Day Total	**153**		
	Cumulative	**4320**		
121	4321-4328	8	**5/5/95**	**Fri**
122	4329-4343	15	**5/8/95**	**Mon**
123	4344-4356	13	5/9/95	Tue
124	4357-4367	11	5/10/95	Wed
125	4368-4418	51	5/11/95	Thu
	5 Day Total	**98**		
	Cumulative	**4418**		
126	4419-4504	86	**5/12/95**	**Fri**
127	4505-4513	9	**5/15/95**	**Mon**
128	4514-4555	42	5/16/95	Tue
129	4556-4584	29	5/17/95	Wed
	TOTAL	**4584**		

** Indicates discontinuous sequences of IDs for Submit Date. First ID of first sequence shown.
       Bolding indicates breaks in defect submit dates.

APPENDIX B
LOGAIS Chronological Node Failure Count Database
CC: Critical Client Node Failure
NC: Non-Critical Client Node Failure
CS: Critical Server Node Failure
NS: Non-Critical Server Node Failure

Interval	Defect ID	Number	Submit Date	CC	NC	CS	NS
1	4,5,7	3	11/11/94		X		
1	9,10,11,13,50,69,113	7	11/11/94				X
2	122,285,303,304	4	11/12/94				X
3	306	1	11/13/94		X		
4	402,407,479	3	11/14/94		X		
4	324,330,332,336,343,344,345,346,352, 357,363,365,367,378,388,406,451,454	18	11/14/94				X
5	602,670	2	11/15/94		X		
5	519,521,543,612,620,635,695,696	8	11/15/94				X
6	718,727	2	11/16/94		X		
6	760,767,768,828,829,835, 836,840,841,842,843,844,876	13	11/16/94				X
7	890,891,927	3	11/17/94				X
8	948,954,955,956,957,958,978	7	11/18/94				X
9	992,993,996	3	11/19/54				X
10	997,1022	2	11/20/94		X		
10	1014,1024	2	11/20/94				X
11	1038,1039	2	11/21/94		X		
11	1061,1062,1081,1090	4	11/21/94				X
12	1152,1154	2	11/22/94				X
13	1234,1235	2	11/23/94		X		
14							
15	1275,1363	2	11/28/94		X		
15	1315,1317,1321,1324,1351,1352, 1364,1366,1367	9	11/28/94				X
16	1401,1402	2	11/29/94	X			

16	1413	1	11/29/94			X	
16	1377,1378,1379,1393,1423,1426, 1439,1482	8	11/29/94				X
17	1548,1549	2	11/30/94	X			
17	1510	1	11/30/94		X		
17	1511,1535	2	11/30/94				X
18	1569	1	12/1//94			·	X
19	1652,1664,1677,1678	4	12/2/94		X		
19	1665	1	12/2/94				X
20	1703	1	12/3/94		X		
21							
22							
23	1769	1	12/6/94	X			
23	1741,1742,1746,1747,1755	5	12/6/94				X
24							
25							
26							
27							
28	1850	1	12/19/94	·	X		
28	1861	1	12/19/94				X
29	1915	1	12/20/94	X			
29	1888,1889,1891	3	12/20/94		X		
30							
31	1933	1	12/22/94	X			
32							
33							
34							
35							
36							
37	1995,1996	2	1/4/95				X
38	2001	1	1/5/95		X		

#		Count	Date				
39							
40							
41	2157	1	1/10/95		X		
41	2096,2106	2	1/10/95				X
42	2165,2225	2	1/11/95		X		
42	2183,2186	2	1/11/95				X
43	2246,2247	2	1/12/95	X			
43	2251	1	1/12/95			X	
44	2293	1	1/13/95		X		
44	2303,2308,2315,2316	4	1/13/95				X
45							
46							
47							
48							
49	2400,2405	2	1/19/95		X		
50	2421,2424	2	1/20/95	X			
51	2633,2634	2	1/24/95		X		
51	2635,2636,2637,2638	4	1/24/95				X
52							
53	2661,2662,2663,2664	4	1/26/95		X		
54	2641,2644,2645,2669, 2671,2672,2673,3003	8	1/27/95		X		
54	2640,2643,2670,2674, 2675,2676,2783	7	1/27/95				X
55	2450	1	1/30/95		X		
56							
57							
58	2487	1	2/2/95				X
59	2511,2512,2513	3	2/3/95		X		
60							
61	3025,3026,3027,3029	4	2/7/95	X			
62							

63							
64							
65							
66	3140,3141,3142,3143	4	2/14/95		X		
67							
68							
69							
70							
71							
72	3263,3264,3265,3266,3267,3268, 3269,3270,3272,3284,3285,3286, 3292,3293,3295,3296,3297,3304, 3305,3306,3307,3314	22	2/23/95				X
73							
74							
75							
76							
77							
78							
79							
80							
81							
82							
83							
84							
85							
86	3422	1	3/16/95	X			
86	3427	1	3/16/95				X
87							
88	3470	1	3/20/95		X		
89	3515	1	3/21/95				X
90	3540,3545,3546,3550,3563	5	3/22/95		X		
91	3567	1	3/23/95		X		

#	Numbers	Count	Date				
92							
93	3629	1	3/27/95		X		
94							
95							
96							
97							
98	3696	1	4/3/95		X		
99							
100							
101	3740,3746,3755	3	4/6/95				X
102							
103							
104	3845,3856	2	4/11/95		X		
105	3871,3872	2	4/12/95	X			
105	3866,3867,3868	3	4/12/95		X		
106	3900,3901	2	4/13/95		X		
107							
108							
109							
110							
111	3976,3982,3991	3	4/19/95		X		
111	3999	1	4/19/95				X
112	4063,4066	2	4/20/95		X		
113	4081	1	4/21/95		X		
114	4127,4128,4129,4131	4	4/26/95	X			
114	4122	1	4/26/95				X
115							
116							
117							
118							
119							

120	4314,4315	2	5/4/95	X			
121							
122	4329,4330	2	5/8/95		X		
123							
124	4357	1	5/10/95		X		
125	4371,4399,4416,4418	4	5/11/95				X
126	4420,4426,4473,4487	4	5/12/95				X
127							
128							
129							
Total				24	83	2	157

SMERFS Session for Critical Client *Time to Failure* Predictions

Excerpted (i.e., only the essential instructions for this example have been retained) from the history file for critical clients, using ten *5 day* failure count intervals to cover the range 1,50). Comments are in parentheses to distinguish them from SMERFS input and output.

ENTER DESIRED DATA TYPE, OR ZERO FOR A LIST.

     4   **(Specifies Failure Count Data)**

THE AVAILABLE INPUT OPTIONS ARE:

  1 ASCII FILE INPUT

  2 KEYBOARD INPUT

  3 LIST THE CURRENT DATA

  4 RETURN TO THE MAIN PROGRAM

  ENTER INPUT OPTION.

     2

A RESPONSE OF NEGATIVE VALUES FOR THE PROMPT:

"ENTER FAULT COUNT AND TESTING LENGTH", WILL END THE PROCESSING.

(**The count of failures in each *5 day* interval is entered for critical clients. See Appendix B. The following list is the response of SMERFS to the keyboard input. For example, the fourth keyboard entry would be 4 1. Eleven failures total are entered. In SMERFS the interval or "TESTING LENGTH" is always "1". In *this* application, the interval is *5 days*)**

ENTER FAULT COUNT AND TESTING LENGTH.

     0.000000000000000E+000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     0.000000000000000E+000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     0.000000000000000E+000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     4.000000000000000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     1.000000000000000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     1.000000000000000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     1.000000000000000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     0.000000000000000E+000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     2.000000000000000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     2.000000000000000   1.000000000000000

ENTER FAULT COUNT AND TESTING LENGTH.

     -1.000000000000000   -1.000000000000000

ENTER COUNT MODEL OPTION, OR ZERO FOR A LIST.
> 4   **(SCHNEIDEWIND MODEL)**

ENTER **ONE** TO INVESTIGATE FOR THE OPTIMUM S (USING TREATMENT TYPE NUMBER 2); ELSE ZERO TO CONTINUE WITH THE MODEL EXECUTION.
> 1

ENTER RANGE OVER WHICH S SHOULD BE TESTED.
> 1   10   **(1 to 10 5 *day* intervals)**

S	BETA	ALPHA	WLS	MSE-F	**MSE-T**
4	.11488E+00	.22871E+01	.17785E+01	.11561E+01	**.24993E+00**

(SMERFS was able to compute only one value of "s" for these data but the result is good because of the low value of Mean Square Error for *Time to Next Failure* (MSE-T))

ENTER DESIRED MODEL TREATMENT NUMBER, OR FOUR TO TERMINATE MODEL EXECUTION.
> 2   **(This "TREATMENT" discards failure data in the range 1,3 for *this* application)**

ENTER ASSOCIATED VALUE OF S (LESS THAN THE NUMBER OF PERIODS).
> 4

TREATMENT 2 MODEL ESTIMATES ARE:

BETA   .11488E+00

ALPHA   .22871E+01

TOTAL NUMBER OF FAULTS   **.19908E+02**   **(Actual=24. See Appendix B)**

PLUS THOSE SKIPPED   .00000E+00 IN PERIODS 1 THROUGH 3

OF FAULTS REMAINING   .89081E+01   **(Actual=13: 24 total-11 entered)**

WEIGHTED SUMS-OF-SQUARES
BETWEEN PREDICTED AND
OBSERVED FAULTS   .17785E+01

MEAN SQUARE ERROR FOR
CUMULATIVE FAULTS   .11561E+01

MEAN SQUARE ERROR FOR
TIME TO NEXT FAILURE   **.24993E+00**

THE AVAILABLE FUTURE PREDICTIONS ARE:

1) THE NUMBER OF FAULTS EXPECTED IN THE NEXT TESTING PERIOD

2) THE NUMBER OF PERIODS NEEDED TO DISCOVER THE NEXT M FAULTS

ENTER PREDICTION OPTION, OR ZERO TO END PREDICTIONS.
    2

ENTER VALUE OF M (BETWEEN ONE AND .89081E+01), OR ZERO TO END.
    1.000000000000000   **(One failure)**
    # OF PERIODS EXPECTED   .10365E+01   **(1.037 intervals = *5.19 days*)**

ENTER VALUE OF M (BETWEEN ONE AND .89081E+01), OR ZERO TO END.
    2.000000000000000   **(Two failures)**
    # OF PERIODS EXPECTED   .22133E+01   **(2.213 intervals = *11.07 days*)**

ENTER VALUE OF M (BETWEEN ONE AND .89081E+01), OR ZERO TO END.
    .3.000000000000000   **(Three failures)**
    # OF PERIODS EXPECTED   .35745E+01   **(3.575 intervals = *17.88 days*)**

ENTER VALUE OF M (BETWEEN ONE AND .89081E+01), OR ZERO TO END.
    4.000000000000000   **(Four failures)**
    # OF PERIODS EXPECTED   .51886E+01   **(5.189 intervals = *25.95 days*)**

ENTER VALUE OF M (BETWEEN ONE AND .89081E+01), OR ZERO TO END.
    5.000000000000000   **(Five failures)**
    # OF PERIODS EXPECTED   .71719E+01   **(7.172 intervals = *35.86* days)**

Edited Computation of Probability of System Failure, Using Statgraphics, for Failure Scenario of Table 7

Psys=((1-(1-Pcc)^Ncc)*((Pnc)^Nnc))+((1-(1-Pcs)^Ncs)*((Pns)^Nns)) (**Probability of System Failure: Equation 12**)

Pcc GETS .005929   (**Load Probability of Critical Client Failure. See Table 3.**)

Ncc GETS 5   (**Load Number of Critical Clients. See Figures 1 and 2.**)

Pnc GETS .02025   (**Load Probability of Non-Critical Client Failure. See Table 3.**)

Nnc GETS 5 5 4 3 2 1 0 (**Load Vector of Non-Critical Clients for Failure Scenario of Table 7**)

Pcs GETS .000494   (**Load Probability of Critical Server Failure. See Table 3.**)

Ncs GETS 1   (**Load Number of Critical Servers. See Figures 1 and 2.**)

Pns GETS .039032   (**Load Probability of Non-Critical Server Failure. See Table 3**)

Nns GETS 1 0 0 0 0 0 0 (**Load Vector of Non-Critical Servers for Failure Scenario of Table 7**)

EVAL **Psys**   (**Compute Probability of System Failure for Failure Scenario of Table 7. These results are entered in Table 9.**)

0.0000192819   0.000494   0.000494005   0.000494243   0.000506013   0.00108723   0.0297895

DISTRIBUTION LIST

Agency	No. of Copies
Defense Technical Information Center 8725 John J. King Rd., STE 0944 Ft. Belvoir, VA 22314	2
Dudley Knox Library, Code 013 Naval Postgraduate School Monterey, CA 93943	2
Office of Research Administration, Code 91 Naval Postgraduate School Monterey, CA 93943	1
Department of Systems Management Library, Code SM/Eb Naval Postgraduate School 555 Dyer Rd Rm 239 Bldg. 330 Monterey, CA 93943	1
Commanding Officer Marine Corps Tactical Systems Support Activity Box 555171 Camp Pendleton. CA 92055-5171	1
Capt. Kenneth Warburton Marine Corps Tactical Systems Support Activity Box 555171 Bldg. 31345 Camp Pendleton, CA 92055-5171	5
Dr. Norman F. Schneidewind Naval Postgraduate School Code SM/Ss Monterey, CA 93943	10